Instituto Tecnológico de Costa Rica.

Ingeniería en computación

Escuela de Computación - Sede Cartago.

Inteligencia Artificial.

II Semestre 2018.

Proyecto 2: Relaciones etimológicas.

Profesor: Juan Esquivel.

Estudiantes:


```
Fernanda Alvarado.
Freyser Jímenez.
Minor Sancho.
```


Sección 1. Introducción.

La derivación lógica es un algoritmo utilizado en Inteligencia Artificial para obtener la solución más óptima de un problema,el propósito de este proyecto es procesar un conjunto de datos por medio de derivación lógica. Los datos que se van a procesar son relaciones etimológicas entre palabras de diferentes idiomas; estos datos se extraen de la base de la base de datos “Etymological Wordnet”, la cual se basa en basada en en.wiktionary.org que recopila relaciones entre palabras en múltiples idiomas (aunque iniciando desde inglés). Tomando estos datos, se puede construir un sistema que, basado en proposiciones lógicas, nos permita responder preguntas relacionadas con orígen común de palabras, similitud entre lenguajes, etc. El proyecto se va a desarrollar con la biblioteca pyDatalog, la cual es una adaptación de un lenguaje lógico en alto nivel y actualmente es muy utilizado en esta rama para realizar multiples proyecto, esto se debe a que es fácil de aprender y se adapta a las necesidades de los desarrolladores.

Sección 2. Descripción de instalación.

Para instalar este proyecto:

1. Descargas la base de datos .tsv: http://www1.icsi.berkeley.edu/~demelo/etymwn; es importante que la base de datos tenga el nombre “etymwn.tsv”
2. Descargar el proyecto https://github.com/ferAlvarado/Proyecto2.
3. Una vez descargado el proyecto se debe descomprimir en el directorio que desee.

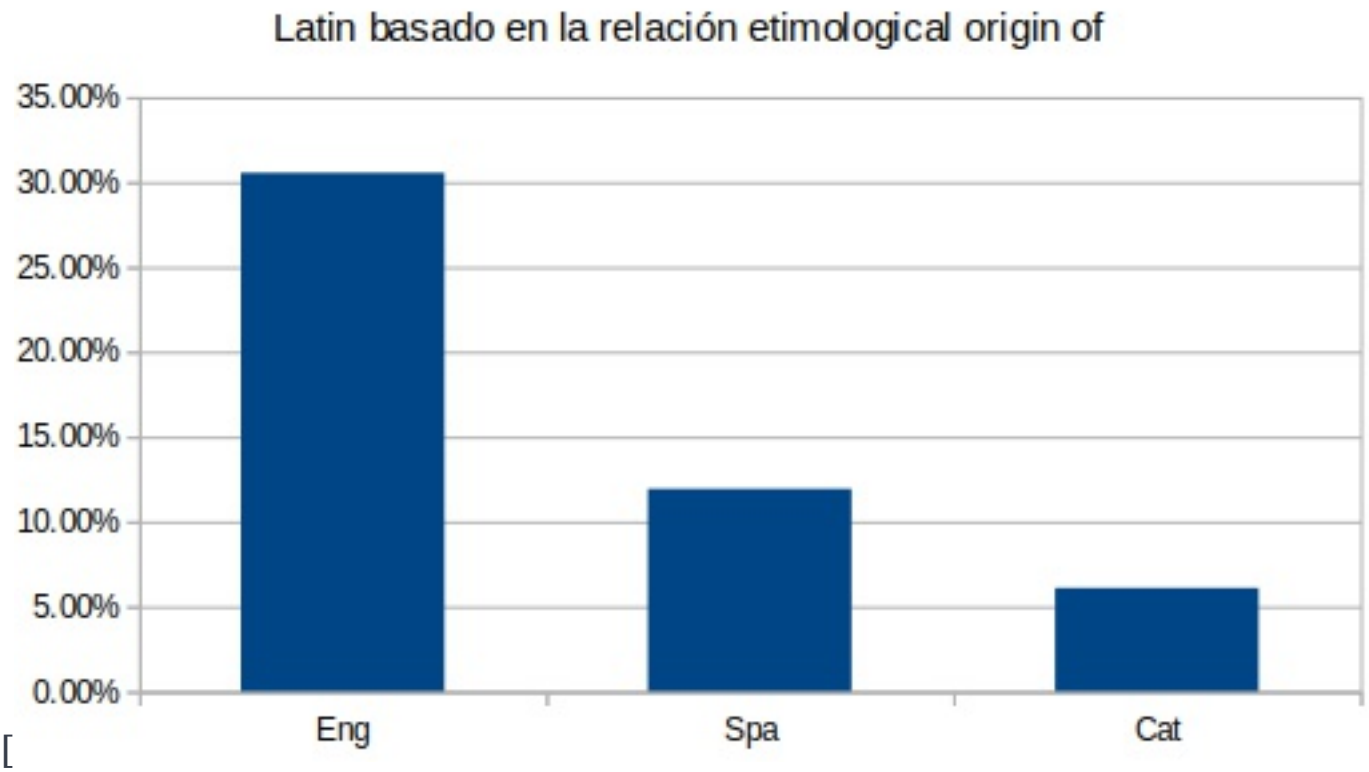
4. Para que el proyecto funcione correctamente, la base de datos ddescargada se debe copiar en la carpeta donde se ejecuta el proyecto; en este caso: /tec/ia/p2/g02.
5. Ejecutar el archivo:

Sección 3. Manual de usuario.

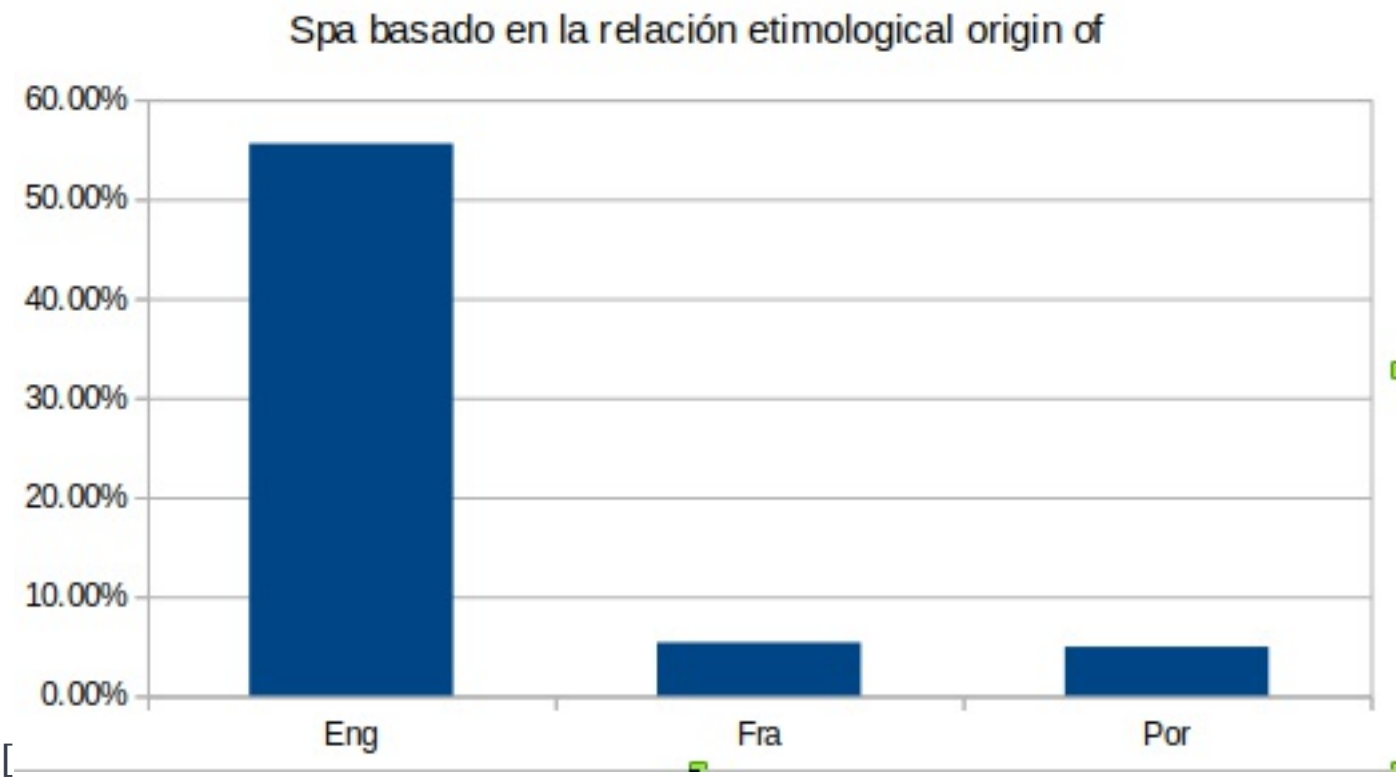
Manual de usuario:https://github.com/ferAlvarado/Proyecto2/blob/master/Manual de instalación.md

Sección 4. Resultados.

Según los datos de registrados en la base y las pruebas realizadas para los diferentes idiomas con etymological origin of se puede deducir el resultado más relevante es el aporte que tiene el español y el latín; tomando en cuenta que el 60% con una relación etimological origin of tiene un aporte del mismo idioma. Además, se observá que el latín y el español tienen una relación muy fuerte ya que el porcentaje de aporte es muy similar.
A continuación, se muestran gráficas de los aportes del latín y el español a otros idiomas:



[



[

Sección 5. Detalles de implementación.

Manejo de datos.

Los datos que utiliza el código desarrollado son los de la base de datos “Etymological Wordnet”, este es un archivo .tsv; para trabajar primero se lee el archivo; una vez realizada esta tarea, se almacenan en una lista que los separa según la relación; esto permite que la búsqueda sea ms eficiente ya que cada relación es la “llave primaria” de cada sublista.
Para leer el archivo se utiliza la siguiente función:

```
def leer(nombre):
    lista = []
    etymology_list=[]
    etymological_origin_list=[]
    has_derived_form_list=[]
    is_derived_from_list=[]
    etymologically_related_list=[]
    derived_list=[]
    variant_orthography_list=[]
    etymologically_list=[]
    with open(nombre,"r",encoding= 'utf-8') as f:
        f = csv.reader(f,delimiter="\t")
        for line in f:
            if line[1] == "rel:etymology" :
                etymology_list += [buscar_dos_puntos_mayor_re
ndimiento(line)]
            elif line[1] =="rel:etymological_origin_of" :
                etymological_origin_list += [buscar_dos_punto
```

```
s_mayor_rendimiento(line)]
            elif line[1] =="rel:has_derived_form" :
                has_derived_form_list += [buscar_dos_puntos_m
ayor_rendimiento(line)]
            elif line[1] =="rel:is_derived_from" :
                is_derived_from_list += [buscar_dos_puntos_ma
yor_rendimiento(line)]
            elif line[1] =="rel:etymologically_related" :
                etymologically_related_list += [buscar_dos_pu
ntos_mayor_rendimiento(line)]
            elif line[1] =="rel:derived" :
                derived_list += [buscar_dos_puntos_mayor_rend
imiento(line)]
            elif line[1] =="rel:variant:orthography" :
                variant_orthography_list += [buscar_dos_punto
s_mayor_rendimiento(line)]
            elif line[1] =="rel:etymologically" :
                etymologically_list += [buscar_dos_puntos_may
or_rendimiento(line)]
            else:
                print("Opcion no encontrada")
                continue
    lista += [etymology_list, variant_orthography_list, deriv
ed_list, etymologically_related_list,
              is_derived_from_list, has_derived_form_list, et
ymological_origin_list,etymologically_list]
    return lista
```

La lista que almacena los datos leídos es la siguiente:

```
lista += [etymology_list, variant_orthography_list, derived_list, etymologically_related_list,
          is_derived_from_list, has_derived_form_list, etymological_origin_list,etymologically_list]
```

Estructuras de control.

Para construir las estructura utilizadas en el proyecto se utilizó la biblioteca pyDatalog; y se hace de la siguiente manera; cada una es una función lógica:

```
from pyDatalog import pyDatalog

pyDatalog.create_atoms('idioma_aportando,contar_palabras,padre,hermano,tio,primo,hijo,palabra_idioma,idioma_palabra,contar_palabras_comunes,listar_palabras_comunes,ancestro,descendiente,idioma,palabra_comun,P,X,Y,A,B')
```

La principal estructura lógica que se implementó es padre; para formar esta estructura se basa en las relaciones etimológicas; es decir dependiendo de las relaciones va a asociar una palabra con otra palabra; donde una de esas palabras es el padre. A continuación se muestra la función padre:

```
def cargarPadres(opciones, datos):
    for palabra in opciones:
        if palabra =="etymological_origin_of":
            for i in datos[6]:
                + padre(i[5],i[1])
        if palabra =="has_derived_form" :
            for i in datos[5]:
                + padre(i[5],i[1])
        if palabra =="etymologically":
            for i in datos[7]:
                + padre(i[5],i[1])
        if palabra == "etymology":
            for i in datos[0]:
                + padre(i[1],i[5])
        if palabra == "variant:orthography":
            for i in datos[1]:
                + padre(i[1],i[5])
        if palabra == "derived":
            for i in datos[2]:
                + padre(i[1],i[5])
        if palabra == "etymologically_related":
            for i in datos[3]:
                + padre(i[1],i[5])

        if palabra == "is_derived_from":
            for i in datos[4]:
                + padre(i[1],i[5])
```

```
    return
```

Además, se declaran otras estructuras: hijo, tío, primos, hermano que se basan en padre; la estructura hermano es utilizada para averigüar si una palabra es prima de otra.

```
hermano(X,Y) <= padre(X,P) & padre(Y,P)  & (X!=Y)

#================HIJO================
#hijo es X y padre es Y
hijo(X,Y) <= padre(X,Y)

#================TIO================
#primero X = sobrino y segundo Y = tio
tio(X,Y) <= padre(X,P) & hermano(Y,P)

#================PRIMO================
# X = primo 1 y Y = primo 2
primo(X,Y) <= padre(X,A) & tio(Y,A)

#================Descendiente================
#A MAYOR Y B MENOR
descendiente(B,A) <= padre(B,A)
descendiente(B,A) <= padre(B,P) & descendiente(P,A)

#================Ancestro================
#A MAYOR Y Y MENOR
```

```
ancestro(A,Y) <= descendiente(Y,A)
```

Sin embargo, esa estructura es la base para realizar las operaciones de dos palabras; para las otras operaciones de igual manera se utiliza padre pero se le adiciona una estructura idioma la cual se encarga de asociar una palabra con un idioma y a partir de esos datos se puede obtener los porcentajes de los idiomas que aportan más.

```
def cargar_Idiomas1(opciones, datos, valor):
    for palabra in opciones:
        if palabra =="etymological_origin_of":
            for i in datos[6]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra =="has_derived_form" :
            for i in datos[5]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra =="etymologically":
            for i in datos[7]:
                if valor==1:
                    + idioma(i[1],i[0])
```

```
                else:
                    + idioma(i[5],i[4])
        if palabra == "etymology":
            for i in datos[0]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra == "variant:orthography":
            for i in datos[1]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra == "derived":
            for i in datos[2]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra == "etymologically_related":
            for i in datos[3]:
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
        if palabra == "is_derived_from":
            for i in datos[4]:
```

```
                if valor==1:
                    + idioma(i[1],i[0])
                else:
                    + idioma(i[5],i[4])
    print("Idiomas Cargados...")
    return
```

Distribución de trabajo.

```
-------------------------------------------------------
-           Nombre          -     % Aporte            -
-------------------------------------------------------
-     Fernanda Alvarado     -         100             -
-------------------------------------------------------
-      Freyser Jímenez      -         100             -
-------------------------------------------------------
-       Minor Sancho        -         100             -
-------------------------------------------------------
```

Link del git: https://github.com/ferAlvarado/Proyecto2